Handling the crowd avoidance problem in job recommendation systems integrating FoDRA

Nikolaos D. Almalis* and George A. Tsihrintzis

Department of Informatics,
University of Piraeus,
Piraeus, Greece
Email: nikosalmalis@unipi.gr
Email: geoatsi@unipi.gr
*Corresponding author

Ioannis Papaioannou

National Technical University of Athens,
Zografou Campus 9, IroonPolytechnioustr,
15780 Zografou, Greece
Email: ipapaioannou@corelab.ntua.gr

Abstract: In this article, we present the basic principles and approaches of job recommender systems (JRSs). Furthermore, we describe the four different relation types of the job seeking and recruiting problem, derived directly from the formal definition of JRSs. We use our previously published four dimensions recommendation algorithm (FoDRA) to calculate the suitability of a person for a job and then we model a job seeking and recruiting problem consisting of many candidates and many jobs (N-N case). Finally, we test the algorithm and present the results proposing a solution – *the minimum acceptable suitability level* – for the crowd avoidance problem that occurs. Our study produces good results and shows that this approach can be considered as an important asset in the domain of job seeking and recruiting.

Keywords: recommender system; job seeking and recruiting; job recommender; information filtering.

Reference to this paper should be made as follows: Almalis, N.D., Tsihrintzis, G.A. and Papaioannou, I. (2020) 'Handling the crowd avoidance problem in job recommendation systems integrating FoDRA', *Int. J. Computational Intelligence Studies*, Vol. 9, Nos. 1/2, pp.128–145.

Biographical notes: Nikolaos D. Almalis holds an MSc in Computer Science from the University of Piraeus (2010) with honours. He graduated from the Greek Military Academy (1999) specialised in IT technologies in military environment. He is a PhD candidate and his primary research interests are in the areas of recommender systems and cryptography. Since 2010, he is a Project Manager in Application Development in the IT Centre of Greek Army.

George A. Tsihrintzis is a Full Professor in the University of Piraeus, Greece and the Head of its Department of Informatics. He received his Diploma of Electrical Engineer from the National Technical University of Athens, Greece (with honours) and MSc and PhD in Electrical Engineering from the Northeastern University, Boston, Massachusetts, USA. His current research

interests include pattern recognition, machine learning, decision theory, and statistical signal processing and the applications of the above in multimedia interactive services, user modelling, knowledge-based software systems, human-computer interaction and information retrieval. He has authored or co-authored over 300 research publications in these areas, which include five monographs and 16 edited volumes. He is the recipient/co-recipient of three best paper awards and he has served as keynote speaker in international conferences. He is the Editor-in-Chief of the *International Journal of Computational Intelligence Studies* (Inderscience) and the *Intelligent Decision Technologies Journal* (IOS Press).

Ioannis Papaioannou is a PhD candidate in the field of Cryptography and Secure Distributed Computing in the School of Electrical and Computer Engineering of the National Technical University of Athens (NTUA). He obtained his MEng in the Computer Engineering and Informatics Department of the University of Patras (2015) and has worked as a Software Engineer in both industry and academia. His primary research interests are in the areas of cryptography, distributed algorithms and block chain technologies.

1 Introduction

In the current era, *information overload* intrudes in many aspects of our lives. This deluge of information makes everyday tasks a challenge. Finding useful or suitable items or objects (such as newspapers, websites, songs, movies, books or even jobs) has become impractical. For this reason, we observe a vast increase in the number of applications that are broadly developed and support human decision making via suggesting services, products and many-sided information to customers (Shani and Gunawardana, 2011; Lampropoulos et al., 2014).

The bulk of modern organisations currently use a mixture of resources to achieve their goals. Although the strategic aims of these organisations can generally vary, they all exploit the same type of resources (such as people, money, facilities and information) to fulfil their objectives. Counter to the other resource types, the management of information has often been neglected due to the difficulty of manipulation of the huge amount of data at their disposal (Das et al., 2007).

The advance of informatics made accessible huge amounts of information in our everyday lives. This increase in the availability of applicable information coupled with significant decrease in the time needed to retrieve it (Kowalski and Maybury, 2006). A common theme of the information age is one of drowning in a sea of data whilst being starved of the necessary information to support decision making (Hall and Slembrouck, 2007).

To handle such obstacles, we use recommender systems (RS), a scientific domain of existing well-established disciplines such as machine learning, artificial intelligence, information retrieval, data mining and human computer interaction (Ricci et al., 2010). RS were created to handle problems and balance the difficulties that appear due to *information overload*. They are tools that utilise numerous techniques and algorithms to insulate unrelated information from large amounts of data and create personalised

suggestions that the appropriate user can examine effectively in a reasonable amount of time, enabling decision making (Burke, 2007; Lampropoulos and Tsihrintzis, 2015).

RS were introduced as a term by Resnick and Varian in 1997 (Resnick and Varian, 1997) and since then invaded several areas of our life (Sotiropoulosand Tsihrintzis, 2017b; Sotiropoulos et al., 2008; Burke, 1999; Linden et al., 2003). In a nutshell, two basic principles characterise the RSs research (Burke et al., 2011):

- 'A recommender system is personalised. The recommendations it produces are meant to optimise the experience of one user, not to represent group consensus for all.'
- 'A recommender system is intended to help the user select among discrete options. Generally the items are already known in advance and not generated in a bespoke fashion.'

Another direct application of RSs exists in the domain of job seeking and recruiting. In the last ten years, the so-called job recommender systems (JRSs) draw the attention and the interest of corporations (Jobvite, 2017) and researchers (Sotiropoulos and Tsihrintzis, 2017a), and radically changed the way that job seeking websites operate. The problem of matching people and jobs demands flexibility and special manipulation and JRSs' objective is to find the most suitable candidate for the posted job and not just a similar one. Suitability is far from equivalent with similarity. For example, a candidate having six or seven years of experience is probably more suitable for a job position that requires at least five years of past working experience than a candidate with exactly five years. Also a job recruiter might prefer other attributes being under a certain limit (i.e., age) (Almalis et al., 2015b).

Although many recommendation approaches tried to solve the above problems (as outlined in Section 2), the quality of their results still remains a challenge. So far, the recommendation results are based on finding similarities based on a specific (Adomavicius and Tuzhilin, 2011; Guo et al., 2014) or Boolean value (Koren et al., 2009; Al-Otaibi and Ykhlef, 2012a). It is clear that we need a new approach that takes into consideration not only the similarity of an exact value but also the diversity of the categories and the measurable skills that depend on a vast range of values.

Another challenge that we need to face is the concept of 'crowd-avoidance' (Gualdi et al., 2013; Zhang et al., 2017). There are situations where the results of our standard RS are poor. For example, when we must recommend a list of hotels to the users planning their vacations in the same place, there is a possibility that a hotel that is more convenient and higher-ranked to the users will be the same. That will result in a no vacancy hotel that will be unable to accommodate the full number of users. The crowd avoidance problem can be very restricting in the case of job recruiting. In the case when many candidates search for many job positions, a skilful candidate can be recommended for more than one job. That could be a serious problem because an employee is an entity that cannot be shared among many positions. We address this problem when we present our experimental results in Sections 3 and 4.

The remainder of the paper is organised as follows: in Section 2 we present an overview of various types of recommendation techniques and previous related research. In Section 3, we describe the relation types between people and jobs based on the formal definition of JRSs. Also, we propose a solution to the crowd-avoidance problem that

occurs especially in the *many to many* (N-N) relation type. Section 4 contains the comparative evaluation of our experimental results. Finally, in Section 5, we conclude the paper and point to future research in the job seeking and recruiting domain.

2 Related work

2.1 Basic recommendation algorithms

The fundamental principles of RS are the following:

1 the recommendations are tailor-made for every final user

2 cooperation with the user to support decision making.

These principles define the theoretical background and guide the technical implementation of RSs. The four different types or RS (Ramezani et al., 2008) that are briefly illustrated below are the strategies that allow us to implement these RSs:

- *Content-based recommenders (CBRs):* using an evaluated set of characteristics for a series of items that are rated by the user CBR constructs the interests-based profile on the features of the items. Using item-based similarity the user profile is matched with the set of characteristics of an item (Lops et al., 2011; Ricci et al., 2011).
- *Collaborative filtering recommenders (CFRs):* CFR is the most favoured technique as it is the most used one. CFR uses social knowledge and it works due to user-based similarity. It find users with relevant or almost relevant interests as the target user and makes recommendations according to their loved items (Sarwar et al., 2001; Ekstrand et al., 2011).
- *Knowledge-based recommenders (KBRs):* these systems utilise the deep understanding of the domain and their principles are based on functional knowledge that has been invented by patterns and rules of the domain. They have the edge on enhancing reliability as they generally incorporate less noise. Their drawback is that they require substantially more knowledge acquisition for installation and maintenance during their working period (Carrer-Neto et al., 2012; Bobadilla et al., 2013).
- *Hybrid recommenders (HRs):* all of the previous techniques have constraints, such as 'cold start' (the condition where the system cannot make any interpretations for users or items when not enough information is collected) or the data sparsity problem (where there are not enough available ratings for the system to use) are some of those limitations (Feil et al., 2016). To solve such problems researchers created hybrid systems that combine two or even more techniques, using one as a base and another as secondary (Burke, 2002;Ghazanfar and Prugel-Bennett, 2010).

2.2 Job recommendation systems

A plethora of recommendation techniques have been designed to deal with the problem of job seeking and recruiting, specifically via internet websites. Recommendation systems had become a crucial and unavoidable part in the process of matching the

appropriate job with the appropriate candidate due to the uninterrupted rise of the amount of available information.

Enachescu (2016) utilised semantic web ontologies and managed to automatically match job positions with user profiles and clarifying and reducing the complexity of the recommendation procedure. The system was implemented using a Java-based platform.

Hong et al. (2013) developed an online job recommendation system called iHR that allocate users into groups using individual data and the behavioural history of the users. This can give us the edge in cases where different users may have different attributes and would not be appropriate to use a single recommendation path.

Nguyen et al. (2016) designed a system that uses adaptive methods to generate job recommendations. Their system applies to a group of candidates exploiting user clustering and selecting suitable methods for each cluster through empirical evaluation.

Paparrizos et al. (2011) proposed a model in order to forecast the next wanted position of a candidate seeking for a job, using machine learning and utilising the candidate's previous work experience and information from other job seekers with related working profiles on the internet.

Al-Adrousy et al. (2014) used a variation of the Delta-SimRank algorithm to help students select their courses. Through their hybrid recommender framework, students could connect to real life business experts and ask for their opinion. As a result, the selection process was significantly reduced and users had increased performance.

Yu et al. (2011) designed an interaction history algorithm using similarity calculations, containing both the candidate's and position recommendation. Extracting useful information from the candidate's resume and then using the precise preferences of the candidates allows the algorithm to retrieve the similarity calculation.

Also, in order to connect efficiently the job market with graduated students, Liu et al. (2016) developed a framework that models a student using his academic and personal features and utilises similarity mechanisms to generate their recommendations.

As we already mentioned, the previous systems and methods attempt to discover correlations based on a specific value. Versatile RS that can handle multiple job requirements are scarce. Seeing that no method can offer perfectly flawless personalisation, it is of paramount importance to collect information about the candidate's preferences and the recruiter's need for the posted job in the form of 'specific value', 'range with upper or lower limit', 'range with upper and lower limit' (Almalis et al., 2015a).

Except from this shortfall of the existing techniques, there is also another obstacle to be addressed, namely the *crowd problem*. In many domains, some researchers call it *crowd wisdom* and it could be wisdom in job seeking and recruiting domain only in some cases (i.e., 1-N and N-1) which we handled well in a previous work of ours (Almalis et al., 2014). But in the N-N case, the crowd problem arises significantly as it is common for many suitable job seekers to be found for one job, but no one for some other.

These limitations motivated and guided our research. In our previous work, we described the problem of job seeking in a formal way and proposed a high-level architecture of a constrain-based JRS using the four dimensional recommendation algorithm (FoDRA) (Almalis et al., 2015b). In this article, we also use FoDRA to compute suitability, and we propose the introduction of the *minimum acceptable*

suitability level (Section 3) to deal with the crowd avoidance problem which appears in *many to many (N-N)* types of job seeking and recruiting.

3 Matching people and jobs

3.1 Entities relations

So far in the literature of JRSs, various titles have been used to depict the research on the aforesaid domain, e.g., *matching jobs and people*, *job seeking*, *job recruiting*. Meanwhile common RSs use two entities: the *user* and the *item*, in order to generate recommendations to the final users. JRs, although they follow this common pattern of RS, they have some special characteristics of their own. These special characteristics depend on the central theme of the title depiction which relies on the relation between the *item* and the *user*. Additionally the relation type between the *item* and the *user* derives from what is suggested to whom. Thus, in a JRS the *item* could be the *job position* when the *user* is a person who seeks a job or the *item* could be the *candidate employee* when the *user* is a *job recruiter* who searches a suitable person to be hired.

All these generic approaches were summarised in our previous work (Almalis et al., 2015b) and they were formulated in a description which is the concrete and formal definition of the job seeking and recruiting problem:

> "Let J be the set of all jobs and let P be the set of all possible candidate employees/job seekers (people) that can be recommended. Let f be a utility function, which measures the suitability of a candidate employee P for a job J that is f: $J \times P \rightarrow R$, where R is a totally ordered set (for example, real numbers within a certain range). For each job $j \in J$ we want to choose such a candidate employee/job seeker $p' \in P$ that maximizes the job's suitability".

Experimenting with the above definition, we could easily produce the different instances for the problem. When the sets J and P are singletons, we have the case of one candidate seeking for one job (1-1). In the case where only one of the two sets is singleton, we have the problem of one candidate and several available jobs (1-N) or more than one candidate and one job available (N-1). Finally, in the case that neither of the two sets are singleton, we have the problem of many job seekers and many available jobs (N-N). To summarise, in order to categorise the problem instances, we proposed four types, which are derived directly from our definition

1 one candidate ↔ one job (1-1)

2 one candidate ↔ many jobs (1-N)

3 many candidates ↔ one job (N-1)

4 many candidates ↔ many jobs (N-N).

3.2 Many candidates ↔ many jobs complexity

Handling the case *many candidates ↔ many jobs* is not so straightforward as the other three cases which are presented in the previous paragraph:

1 one candidate ↔ one job (1-1)

2 one candidate ↔ many jobs (1-N)

3 many candidates ↔ one job (N-1) and have been dealt with satisfaction in our preview work using FoDRA.

In the *many candidates* ↔ *many jobs* case there is a discrete difficulty in producing recommendations. The recommendations are related to the matching of each person of the set of the candidates/job seekers with a job of the set of jobs. This procedure is completed when all the items of the first set have been matched with the items of the second set. In case one of the two sets have different quantities of items, then the procedure is completed when the items of one of the sets has completely been matched. Finally the end of the aforementioned procedure defines a permutation of the items from the jobs' set and candidates' set and this is one-only one-recommendation on our matching problem.

In order to make much more understandable the difficulty and the complexity in producing recommendations on the case of *many candidates* ↔ *many jobs*, we present a simple but meaningful example. Let us assume two sets with the same number of items. Let be a set J of three jobs (job_1, job_2, job_3) and a set C of three candidates employees (can_1, can_2, can_3). Having no other information about the items of the sets we want to make recommendations. On Table 1 are depicted all the possible permutations of our problem, while on Table 2 we can see all the possible recommendations which are the *Cartesian product* of the two assigned sets. The number of the recommendations equals $n!$, where n is the number of the items. Thus in our example, Table 2 shows six recommendations, as 3! = 6. Similarly, if the sets have four items, the number of recommendations rises to 4! = 24.

Table 1 All possible combinations between equal number of jobs and candidates

		Jobs		
		job_1	job_2	job_3
Candidates	can_1	$job_1 - can_1$	$job_2 - can_1$	$job_3 - can_1$
	can_2	$job_1 - can_2$	$job_2 - can_2$	$job_3 - can_2$
	can_3	$job_1 - can_3$	$job_2 - can_3$	$job_3 - can_3$

Also, we have to clarify that in case the sets have different number of items the complexity is the same as the case of the same number. For example, if the set of the jobs has three items and the set of candidates one item less that means two items, the possible combinations and the possible recommendations are illustrated on Tables 3 and 4 and are calculated as in the same number sets case. As it can be seen by Table 4 results in each recommendation one job is without staff. Also in this second example, for a better view, in the place of the inexistent item we use the word *NONE*.

The previous two examples are useful in describing the complexity of the *many candidates* ↔ *many jobs* relation in finding recommendations. Meanwhile through a strict perceptive of RS basic principles, i.e., the personalisation and the usefulness of the

results, the recommendations of Tables 2 and 4 are not actually recommendations but rather an exhaustive search of all possible options.

Table 2 All possible recommendations according to Table 1 data

	Recommendations		
1st	$job_1 - can_1$	$job_2 - can_2$	$job_3 - can_3$
2nd	$job_1 - can_1$	$job_2 - can_3$	$job_3 - can_2$
3rd	$job_1 - can_2$	$job_2 - can_1$	$job_3 - can_3$
4th	$job_1 - can_2$	$job_2 - can_3$	$job_3 - can_1$
5th	$job_1 - can_3$	$job_2 - can_1$	$job_3 - can_2$
6th	$job_1 - can_3$	$job_2 - can_2$	$job_3 - can_1$

Table 3 All possible combinations between unequal number of jobs and candidates

		Jobs		
		job_1	job_2	job_3
Candidates	can_1	$job_1 - can_1$	$job_2 - can_1$	$job_3 - can_1$
	can_2	$job_1 - can_2$	$job_2 - can_2$	$job_3 - can_2$
	NONE	$job_1 - NONE$	$job_2 - NONE$	$job_3 - NONE$

Table 4 All possible recommendations according to Table 3 data

	Recommendations		
1st	$job_1 - can_1$	$job_2 - can_2$	$job_3 - NONE$
2nd	$job_1 - can_1$	$job_2 - NONE$	$job_3 - can_2$
3rd	$job_1 - can_2$	$job_2 - can_1$	$job_3 - NONE$
4th	$job_1 - can_2$	$job_2 - NONE$	$job_3 - can_1$
5th	$job_1 - NONE$	$job_2 - can_1$	$job_3 - can_2$
6th	$job_1 - NONE$	$job_2 - can_2$	$job_3 - can_1$

Based on the basic principles of RSs, we update and improve the produced so called recommendations of the aforementioned *many candidates* ↔ *many jobs* relation, proposing the concept of the *minimum acceptable suitability level* for a job. Moreover we mention that the calculation of the suitability of each candidate for a job is computed using the FoDRA. In the next paragraph we explain our approach.

3.3 'Minimum acceptable suitability level' approach

As we have already mentioned, our aim is to reduce the complexity of the *many candidates* ↔ *many jobs* relation, producing recommendations based on the basic principles of RSs, i.e., personalisation and usefulness of the results. The examples of the previous paragraph show exactly the lack of personalisation, as the results didn't contain any personal preference, and also the lack of usefulness, as the amount of the produced list was too large to be manipulated.

The concept of the *minimum acceptable suitability level* for a job integrates in the final results the two basic principles and transforms them to real recommendations. Through in a third example we explain our proposed concept using FoDRA for the calculation of the suitability of each candidate for a job.

Thus, we assume there are two sets with the same number of items. Let be a set J of three jobs (job_1, job_2, job_3) and a set C of three candidates employees (can_1, can_2, can_3). Also we calculate the suitability S of each candidate for a job. Table 5 shows the summary of the information. To make our concept clearer, FoDRA produces a number which describes the suitability and if it is 0 means that the candidate has exactly the qualification the job requires, if it is positive means that the candidate is high-qualified and if it is negative means that the candidate is low-qualified. Also the bigger the suitability number, the higher-qualified the candidate is. The opposite happens if the number is below 0.

Table 5 All possible combinations between equal number of jobs and candidates and the calculated suitability for each combination using FoDRA

		Jobs		
		job_1	job_2	job_3
Candidates	can_1	$job_1 - can_1$ (S: +1)	$job_2 - can_1$ (S: –1)	$job_3 - can_1$ (S: 0)
	can_2	$job_1 - can_2$ (S: 0)	$job_2 - can_2$ (S: –3)	$job_3 - can_2$ (S: +2)
	can_3	$job_1 - can_3$ (S: +3)	$job_2 - can_3$ (S: +2)	$job_3 - can_3$ (S: –3)

At this point, we define the *minimum acceptable suitability level* for a job, in order to produce personalised and useful recommendations. So, let be the minimum acceptable suitability level for all the jobs the same and equal to 0. To translate this suitability level it means that the acceptable level of suitability is at least that one which meets the requirements of a job. Finally on Table 6 we have the calculated recommendations which satisfy the criteria of the minimum acceptable suitability level and also the recommendations are sorted according to the summary of all the suitability numbers of the current combinations. The first recommendation has a summary of suitability: 5 ([$job_1 - can_1$ (S: +1), $job_2 - can_3$ (S: +2) and $job_3 - can_2$ (S: +2)] = 5) and the second recommendation has a summary of suitability: 2 ([$job_1 - can_2$ (S: 0), $job_2 - can_3$ (S: +2) and $job_3 - can_1$ (S: 0)] = 2). The rest of the combinations (see Table 7) are not recommended because they are outside of the desirable limit.

Table 6 Recommendations according to Table 5 data

	Recommendations		
1st	$job_1 - can_1$ (S: +1)	$job_2 - can_3$ (S: +2)	$job_3 - can_2$ (S: +2)
2nd	$job_1 - can_2$ (S: 0)	$job_2 - can_3$ (S: +2)	$job_3 - can_1$ (S: 0)

In the next paragraph, for the proof of concept we conduct a comparative experimental evaluation using real-world data from the site of Kaggle (http://www.kaggle.com) after we have modelled the items of the job and the candidate_employee, due to the processing needs of FoDRA.

Table 7 Not accepted combinations according to Table 5 data

	Combinations		
1st	$job_1 - can_1$ (S: +1)	$job_2 - can_2$ (S: –3)	$job_3 - can_3$ (S: –3)
2nd	$job_1 - can_2$ (S: 0)	$job_2 - can_1$ (S: –1)	$job_3 - can_3$ (S: –3)
3rd	$job_1 - can_3$ (S: +3)	$job_2 - can_1$ (S: –1)	$job_3 - can_2$ (S: +2)
4th	$job_1 - can_3$ (S: +3)	$job_2 - can_2$ (S: –3)	$job_3 - can_1$ (S: 0)

4 Comparative experimental evaluation

4.1 Experiment methodology and dataset

To illustrate the discrete differences and upgrades on the recommendation procedure of our approach we will conduct a comparative evaluation. Our proposed constrain-based technique will be compared with a standard recommendation technique, the exact/Boolean value (Al-Otaibi and Ykhlef, 2012b). In both instances we have to generate recommendations for candidates that applied for a number of job positions and give solution to any crowd-avoidance problems may occur.

In the beginning (refinement of the algorithm), the human resources director of a publishing company will feed manually the system with the content of the job description. That is literal references of the job requirements, which the applicant qualifications should match. For our example the required skills are the following: education level (S1), *English language knowledge (S2)*, *translation working experience (S3)* and *age (S4)*. Then attribute values and weights are linked with each skill, representing the level of each skill required and the importance of each skill for the specific job. The values are formed based on a table modified to the specifications of the publishing company and both experiments use the same (see Table 8). In the next step the two algorithms are executed with the posted jobs and 60 job seekers as an input. The similarity measures that are used are the Manhattan ($p = 1$) and Euclidian ($p = 2$) distance and from the generated results we retrieve the top choices.

Table 8 Scoring levels for each required skill

Attribute names (required skill)		*Scoring levels*
S1	Education level	BSc-10 pts, MSc-20 pts, PhD-30 pts, postdoc-40 pts
S2	English language knowledge	Intermediate (B1)-5 pts, upper intermediate (B2)-10 pts, advanced (C1)-20 pts, proficiency (C2)-40 pts
S3	Translation experience	2 pts/year of experience
S4	Age	10 pts(25–30), –2pts/year > 30 and –1pt/year < 25

To be able to find similarities between the profiles of the job seekers and the job positions, they must have the same structure. For that reason the produced profiles of the candidates (based on their CVs) are linked to the position's attributes.

In order to be processed, the collected data must be first transformed into a manipulated expression. For that reason a 4 elements (one for each skill) matrix is created to represent each posted job (J). Each of these elements has a value v_i and a weight w_i where $i = 1, 2, 3, 4$ and $p_i = v_i \times w_i$ are the elements of $P = (p_1, p_2, p_3, p_4)$. Correspondingly, a four elements matrix $S = (s_1, s_2, s_3, s_4)$ is created in order to represent the candidates, where each element of the matrix is portraying the same attribute of the posted job.

The novelty of our approach is the way we enforce tie-breakers. In a scenario with many posted jobs and many job seekers, we execute our algorithm for every job seeker and for every job. In such a case, an outcome where a single candidate is optimal and more suitable for multiple positions is probable. To solve such a crowd-avoidance problem we sort the list of the posted jobs by order of importance. When our algorithm proposes the same candidate for more than one position, the position that is more important will be filled first. If a job needs to be filled by a candidate that is no longer available, then by order of importance, the next best applicant will be hired.

In the next section we will present the results of each technique, but before we notice the basic difference between the two frameworks. In the Exact/Boolean value approach the attributes can have only an exact value whereas in our constrain-based approach system, values can have one of the aforestated four forms: exact value (E), range with lower limit (L), range with upper limit (U), range with lower and upper limit (LU).

Table 9 Skills required for each job

Job 1 – attribute names (required skill)		*Attribute values (required skill level)*	*Attribute weights (importance of required skill)*	*Job 3 – attribute names (required skill)*		*Attribute values (required skill level)*	*Attribute weights (importance of required skill)*
S1	Education level	10	1	S1	Education level	20	1
S2	English language knowledge	5	1	S2	English language knowledge	20	1
S3	Working experience	1	1	S3	Working experience	14	1
S4	Age	7	0.8	S4	Age	10	0.8
Job 2 – attribute names (required skill)		*Attribute values (required skill level)*	*Attribute weights (importance of required skill)*	*Job 4 – attribute names (required skill)*		*Attribute values (required skill level)*	*Attribute weights (importance of required skill)*
S1	Education level	20	1	S1	Education level	30	1
S2	English language knowledge	10	1	S2	English language knowledge	10	1
S3	Working experience	10	1	S3	Working experience	10	1
S4	Age	10	0.8	S4	Age	6	0.8

4.2 Exact/Boolean value job recommendation results

Following the exact/Boolean value approach we construct the structured form of the posted positions. Four discrete jobs are presented in Table 9. The first (job 1) is an entry level job, whereas the last (job 4) is a specialist position (perhaps a position that includes translation of ancient texts). Each attribute has an exact value and a specific weight. For example job 3 has the following requirements: education level: (MSc, 20 points), English language knowledge: (advanced: C1, 20 points), working experience: (seven years, 14 points), age: (25 to 30 years, 10 points). Age has a weight of 0.8 and all the rest skills have a weight of 1.

After the execution the top candidates which are most suitable for each of the posted jobs are shown in Table 10 while in Table 11 we see their skills values. Each of the posted jobs requires different skills which are listed below. We must also rank these jobs by order of importance. That is a choice that will be made by the company: job 4 must be manned first and then respectively job 3, job 2 and job 1.

Table 10 Results for each job

Results for job 1				*Results for job 3*			
Euclidian distance		*Manhattan distance*		*Euclidian distance*		*Manhattan distance*	
Candidate ID	*Final score*	*Candidate ID*	*Final score*	*Candidate ID*	*Final score*	*Candidate ID*	*Final score*
59	10.4	*59*	13.4	*66*	5.83	*66*	8
38	10.48	38	13.8	52	8.01	52	12.6
58	10.59	58	14.6	46	9.43	46	13
8	10.78	8	15.4	50	11.1	50	16.4
14	11.5	14	17.4	5	14.94	13	21.8
39	11.5	39	17.4	13	16.18	53	22.6
3	12.24	3	17.8	53	16.25	14	23
53	12.24	53	17.8	3	16.54	39	23
Results for job 2				*Results for job 4*			
Euclidian distance		*Manhattan distance*		*Euclidian distance*		*Manhattan distance*	
Candidate ID	*Final score*	*Candidate ID*	*Final score*	*Candidate ID*	*Final score*	*Candidate ID*	*Final score*
66	5.1	66	6	*5*	0.89	*5*	0.8
13	5.46	*13*	7.8	50	10.04	50	10.8
52	5.67	52	8.6	3	11.36	3	17
53	5.67	53	8.6	52	11.5	52	18.6
14	6.4	14	9	53	11.5	53	18.6
39	6.4	39	9	13	11.67	66	19.2
3	6.47	3	10.2	66	11.78	13	19.4
8	7.81	8	11	14	12.4	58	21.8

Our first assessment is that a lot of the candidates are suitable for the posted jobs. The final scores that represent distances and are interpreted as suitability are all positive numbers. The closer to zero a candidate is the more suitable is for the posted position. If zero is scored, then the candidate meets the position's requirements to the point. Also we notice that the similarity measure does not drastically affect the accuracy of the generated recommendation. The results are almost the same for Manhattan end Euclidian distances, especially in the top candidates.

The final recommendation results for exact/Boolean value are depicted on Table 12. We can easily observe that ID-66 is the most suitable candidate for jobs 2 and 3 and there is a conflict that will be resolved in the way we addressed above. Job 4 will select his employee first and that will be the candidate with ID-5. Job 3 will be filled next: the most suitable candidate is *ID-66*. We continue with job 2: ID-66 is not available at the time, and *ID-13*, the next best available candidate will be selected. Finally, *ID-59* will be positioned to job 1. As we described in Section 3.3 our *best* recommendation has a summary of suitability numbers of 13.58 (using Euclidean distance) and 30 (using Manhattan distance).

Table 11 Candidate skills

Candidate ID	*Education level*	*English language knowledge*	*Working experience*	*Age*
3	20	5	8	6
5	30	10	10	7
8	20	5	4	10
13	20	5	8	9
14	20	5	6	10
27	20	40	10	10
38	20	5	4	8
39	20	5	6	10
46	20	20	6	10
50	20	20	12	7
52	20	15	8	8
53	20	5	8	8
58	20	5	4	5
59	20	5	2	10
66	20	15	11	10

Table 12 Exact/Boolean value job recommendation results

Job position	*Most qualified candidates*	*Final assignment*
Job 1	ID-59	ID-59
Job 2	ID-66	ID-13
Job 3	ID-66	ID-66
Job 4	ID-5	ID-5

4.3 Job recommendation results using minimum acceptable suitability level approach

On the other hand, following the constrain-based approach we construct the structured form of the posted jobs in Table 13. The attributes values have one of the four mentioned forms: exact value (E), range with lower limit (L), range with upper limit (U), range with lower and upper limit (LU). After the execution of the algorithm FoDRA, the top candidates which are most suitable for each of the posted job are shown in Table 14 (the scored skills that are used were based on Table 8).

We notice that more than one candidate meets the requirements of the jobs posted, due to the expressiveness features of FoDRA, in formulating the job-seeking domain. The higher a final score is, the more suitable the job seeker is for the open position. We also conclude that similarity measure does not affect again the prediction accuracy of the procedure.

Table 13 Skills required for each job

Job 1 –attribute names (required skill)		*Attribute values (required skill level)*	*Attribute weights (importance of required skill)*	*Job 3 – attribute names (required skill)*		*Attribute values (required skill level)*	*Attribute weights (importance of required skill)*
S1	Education level (U)	10	1	S1	Education level (U)	20	1
S2	English language knowledge (E)	5	1	S2	English language knowledge (E)	20	1
S3	Working experience (L)	1	1	S3	Working experience (L)	14	1
S4	Age (LU)	7–9	0.8	S4	Age (LU)	9–11	0.8
Job 2 –attribute names (required skill)		*Attribute values (required skill level)*	*Attribute weights (importance of required skill)*	*Job 4 – attribute names (required skill)*		*Attribute values (required skill level)*	*Attribute weights (importance of required skill)*
S1	Education level (U)	20	1	S1	Education level (U)	30	1
S2	English language knowledge (E)	10	1	S2	English language knowledge (E)	10	1
S3	Working experience (L)	10	1	S3	Working experience (L)	10	1
S4	Age (LU)	9–11	0.8	S4	Age (LU)	5–7	0.8

Again, to finally match the candidates with their appropriate jobs we must deal with crowd-avoidance. Jobs are again ranked by importance in the same way we presented above. The final matching will be the following: job 4: *ID-3*, job 3: *ID-13*, job 2: *ID-53*, job 1: *ID-14*. That recommendation has a summary of suitability numbers: 18.7 (in both Euclidean and Manhattan distances). The final recommendation results for constrain-based approach are depicted on Table 15.

Table 14 Results for each job

Results for job 1				*Results for job 3*			
Euclidian distance		*Manhattan distance*		*Euclidian distance*		*Manhattan distance*	
Candidate ID	*Final score*	*Candidate ID*	*Final score*	*Candidate ID*	*Final score*	*Candidate ID*	*Final score*
13	–3	13	–3	3	9	3	9
53	–3	53	–3	*13*	9	*13*	9
3	–3.8	3	–3.8	53	9	53	9
14	–5.8	*14*	–5.8	14	7	14	7
39	–5.8	39	–5.8	39	7	39	7
8	–6.2	8	–6.2	50	6	50	6
38	–7	38	–7	8	5	8	5
58	–8.6	58	–8.6	38	5	38	5
Results for job 2				*Results for job 4*			
Euclidian distance		*Manhattan distance*		*Euclidian distance*		*Manhattan distance*	
Candidate ID	*Final score*	*Candidate ID*	*Final score*	*Candidate ID*	*Final score*	*Candidate ID*	*Final score*
13	3	13	3	*3*	13	*3*	13
53	2.2	*53*	2.2	13	13	13	13
14	1	14	1	53	13	53	13
39	1	39	1	14	11	14	11
3	0.6	3	0.6	39	11	39	11
8	–1	8	–1	50	10	50	10
50	–1.6	50	–1.6	8	9	8	9
38	–1.8	38	–1.8	38	9	38	9

Table 15 Constrain-based job recommendation results

Job position	*Most qualified candidates*	*Final assignment*
Job 1	ID-13	ID-14
Job 2	ID-13	ID-53
Job 3	ID-3	ID-13
Job 4	ID-3	ID-3

5 Conclusions and future work

In this article, we presented the basic principles and approaches of the JRSs. We addressed the entities that are utilised in standard RS and formally described the problem of job seeking. Furthermore, we described the four different types of the job seeking and recruiting problem, derived directly from our definition. We also illustrated briefly our already published FoDRA algorithm and after we modelled a job seeking problem with many candidates and many jobs. Finally, we executed the algorithm and presented the

results proposing a solution – *the minimum acceptance suitability level* – for the crowd avoidance problem that occurred.

Evaluating the results of our experiment, we can conclude that RS' importance in improving the matching quality in the area of job recruiting is undeniable. A coherent and systematic procedure navigates the companies in order to use our algorithm effectively. The benefits that our JRS provides are the following:

a smaller number of job seekers for in depth evaluation

b boost in the internal operation level of the companies

c improvement on the job description of a position

d results can be used for internal personnel evaluation.

Finally, consolidating the significance of our algorithm, we provided a solution for the (N-N) type of the problem, in the manners of crowd avoidance by ranking the importance of each user type entity (job position). As a result, we now have covered and provided a solution for the full spectrum of the instances that derive from the definition of the job seeking problem.

The methodology we proposed presents promising performance and encourages us to further investigate its applications and extensions. We aim to conduct further research to improve performance in terms of reliability and time response. We also need to test our algorithm on other datasets retrieved from branches of the vast array of the corporate sector. Directions and guidelines for further research also include the modification of the job description based on the production level of the chosen candidates and also questions about the scalability of our algorithm on other common interest fields. We already consider this and other relative research and the results will be published in the near future.

References

Adomavicius, G. and Uzhilin, A. (2011) 'Context-aware recommender systems', in *Recommender Systems Handbook*, pp.217–253, Springer, USA.

Al-Adrousy, W.M., Ali, H.A. and Hamza, T.T. (2014) 'A framework for career-education hybrid recommender system using a selective path Delta-SimRank algorithm', *International Journal of Computer Applications*, Vol. 90, No. 2, pp.42–47.

Almalis, N., Tsihrintzis, G. and Karagiannis, N. (2015a) 'A new content-based recommendation algorithm for job recruiting', in *Research and Development in Intelligent Systems XXXII*, Springer, Cham, Nov., pp.393–398.

Almalis, N.D., Tsihrintzis, G.A., Karagiannis, N. and Strati, A.D. (2015b) 'FoDRA – a new content-based job recommendation algorithm for job seeking and recruiting', in *2015 6th International Conference on Information, Intelligence, Systems and Applications (IISA)*, July, pp.1–7, IEEE.

Almalis, N.D., Tsihrintzis, G.A. and Karagiannis, N. (2014) 'A content based approach for recommending personnel for job positions', in *The 5th International Conference on Information, Intelligence, Systems and Applications, IISA 2014*, July, pp.45–49, IEEE.

Al-Otaibi, S.T. and Ykhlef, M. (2012a) 'Job recommendation systems for enhancing e-recruitment process', in *Proceedings of the International Conference on Information and Knowledge Engineering (IKE)*, *The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp)*, January, p.1.

Al-Otaibi, S.T. and Ykhlef, M. (2012b) 'Job recommendation systems for enhancing e-recruitment process', in *Proceedings of Information and Knowledge Engineering Conference. (IKE 2012).*

Bobadilla, J., Ortega, F., Hernando, A. and Gutiérrez, A. (2013) 'Recommender systems survey', *Knowledge-Based Systems*, Vol. 46, pp.109–132.

Burke, R. (1999) 'The Wasabi personal shopper: a case-based recommender system', in *AAAI/IAAI*, July, pp.844–849.

Burke, R. (2002) 'Hybrid recommender systems: survey and experiments', *User Modeling and User-Adapted Interaction*, Vol. 12, No. 4, pp.331–370.

Burke, R. (2007) *The Adaptive Web, Chap. Hybrid Web Recommender Systems*, pp.377–408, Springer, Berlin, Heidelberg, Berlin-Deutschland.

Burke, R., Felfernig, A. and Göker, M.H. (2011) 'Recommender systems: an overview', *Ai Magazine*, Vol. 32, No. 3, pp.13–18, Jobvite, Social Recruitment Survey, 2015.

Carrer-Neto, W., Hernández-Alcaraz, M.L., Valencia-García, R. and García-Sánchez, F. (2012) 'Social knowledge-based recommender system, application to the movies domain', *Expert Systems with Applications*, Vol. 39, No. 12, pp.10990–11000.

Das, A.S., Datar, M., Garg, A. and Rajaram, S. (2007) 'Google news personalization: scalable online collaborative filtering', in *Proceedings of the 16th International Conference on World Wide Web*, May, pp.271–280, ACM.

Ekstrand, M.D., Riedl, J.T. and Konstan, J.A. (2011) 'Collaborative filtering recommender systems', *Foundations and Trends® in Human-Computer Interaction*, Vol. 4, No. 2, pp.81–173.

Enachescu, M.I. (2016) 'A prototype for an e-recruitment platform using semantic web technologies', *Informatica Economica*, Vol. 20, No. 4, p.62.

Feil, S., Kretzer, M., Werder, K. and Maedche, A. (2016) 'Using gamification to tackle the cold-start problem in recommender systems', in *Proceedings of the 19th ACM Conference on Computer Supported Cooperative Work and Social Computing Companion*, February, pp.253–256, ACM.

Ghazanfar, M.A. and Prugel-Bennett, A. (2010) 'A scalable, accurate hybrid recommender system', in *Third International Conference on Knowledge Discovery and Data Mining, 2010.WKDD'10*, January, pp.94–98, IEEE.

Gualdi, S., Medo, M. and Zhang, Y.C. (2013) 'Crowd avoidance and diversity in socio-economic systems and recommendations', *EPL (Europhysics Letters)*, Vol. 101, No. 2, p.20008.

Guo, X., Jerbi, H. and O'Mahony, M.P. (2014) 'An analysis framework for content-based job recommendation', in *22nd International Conference on Case-Based Reasoning (ICCBR)*, Cork, Ireland, 29 September–01 October 2014.

Hall, C. and Slembrouck, S. (2007) 'Professional categorization, risk management and inter-agency communication in public inquiries into disastrous outcomes', *British Journal of Social Work*, Vol. 39, No. 2, pp.280–298.

Hong, W., Zheng, S., Wang, H. and Shi, J. (2013) 'A job recommender system based on user clustering', *Journal of Computers*, Vol. 8, No. 8, pp.1960–1967.

Jobvite (2017) *Recruiter Nation Report 2017* [online] http://www.jobvite.com (accessed 10 November 2017).

Koren, Y., Bell, R. and Volinsky, C. (2009) 'Matrix factorization techniques for recommender systems', *Computer*, Vol. 42, No. 8, pp.30–37.

Kowalski, G.J. and Maybury, M.T. (2006) *Information Storage and Retrieval Systems: Theory and Implementation*, Vol. 8, Springer Science and Business Media, Kluwer Academic Publishers Norwell, MA, USA ©2000.

Lampropoulos, A.S. and Tsihrintzis, G.A. (2015) *Machine Learning Paradigms: Applications in Recommender Systems*, Vol. 92, Springer Publishing Company, Incorporated ©2015.

Lampropoulos, A.S., Sotiropoulos, D.N. and Tsihrintzis, G.A. (2014) 'Cascade hybrid recommendation as a combination of one-class classification and collaborative filtering', *International Journal on Artificial Intelligence Tools*, Vol. 23, No. 04, p.1460009.

Linden, G., Smith, B. and York, J. (2003) 'Amazon.com recommendations: item-to-item collaborative filtering', *IEEE Internet Computing*, Vol. 7, No. 1, pp.76–80.

Liu, R., Ouyang, Y., Rong, W., Song, X., Xie, W. and Xiong, Z. (2016) 'Employer oriented recruitment recommender service for university students', in *International Conference on Intelligent Computing*, August, pp.811–823, Springer International Publishing.

Lops, P., De Gemmis, M. and Semeraro, G. (2011) 'Content-based recommender systems: state of the art and trends', in *Recommender Systems Handbook*, pp.73–105, Springer, USA.

Nguyen, Q.D., Huynh, T. and Nguyen-Hoang, T.A. (2016) 'Adaptive methods for job recommendation based on user clustering', in *2016 3rd National Foundation for Science and Technology Development Conference on Information and Computer Science (NICS)*, September, pp.165–170, IEEE.

Paparrizos, I., Cambazoglu, B.B. and Gionis, A. (2011) 'Machine learned job recommendation', in *Proceedings of the fifth ACM Conference on Recommender Systems*, October, pp.325–328, ACM.

Ramezani, M., Bergman, L., Thompson, R., Burke, R. and Mobasher, B. (2008) 'Selecting and applying recommendation technology', in *International Workshop on Recommendation and Collaboration in Conjunction with 2008 International ACM Conference on Intelligent User Interfaces*, IUI, January, pp.613–620.

Resnick, P. and Varian, H.R. (1997) 'Recommender systems', *Communications of the ACM*, Vol. 40, No. 3, pp.56–58.

Ricci, F., Rokach, L. and Shapira, B. (2011) 'Introduction to recommender systems handbook', in *Recommender Systems Handbook*, pp.1–35, Springer, USA.

Sarwar, B., Karypis, G., Konstan, J. and Riedl, J. (2001) 'Item-based collaborative filtering recommendation algorithms', in *Proceedings of the 10th International Conference on World Wide Web*, April, pp.285–295, ACM.

Shani, G. and Gunawardana, A. (2011) 'Evaluating recommendation systems', *Recommender Systems Handbook*, pp.257–297, Springer Science+Business Media, LLC, Springer, US.

Sotiropoulos, D.N. and Tsihrintzis, G.A. (2017a) 'Artificial immune system-based classification in extremely imbalanced classification problems', *International Journal on Artificial Intelligence Tools*, Vol. 26, No. 03, p.1750009.

Sotiropoulos, D.N. and Tsihrintzis, G.A. (2017b) 'Machine learning paradigms', in *Machine Learning Paradigms*, pp.107–129, Springer International Publishing, Cham, Switzerland.

Sotiropoulos, D.N., Lampropoulos, A.S. and Tsihrintzis, G.A. (2008) 'Artificial immune system-based music genre classification', in *New Directions in Intelligent Interactive Multimedia*, pp.191–200, Springer, Berlin, Heidelberg, Berlin-Deutschland.

Yu, H., Liu, C. and Zhang, F. (2011) 'Reciprocal recommendation algorithm for the field of recruitment', *Journal of Information and Computational Science*, Vol. 8, No. 16, pp.4061–4068.

Zhang, Y.W., Wang, R., Wang, T.W., He, G.D. and Hu, J. (2017) 'An asymmetric graph recommendation algorithm based on user scores and interests', in *IOP Conference Series: Materials Science and Engineering*, September, Vol. 231, No. 1, p.012045, IOP Publishing.